課題 2 レポート

Name: SARMIENTO DIAZ MARCELO JUNIOR 09A18031

The programming language used is Python 3, also I used Jupyter notebbok to run the program.
In addition, python 3 has several libraries. Which I use the following as:

- import numpy as np (This library helped me to easily create vectors of the form np.linespace () or array operations.))
- import matplotlib.pyplot as plt (I used it for various purposes, especially when plotting functions, creating lines, graphics, legends, etc.)
- import math (This library helps me to be able to express mathematical functions as logarithms and exponentials e ^ x)

2-1) メタノール合成反応器に合成ガスを導入して反応させる。以下の反応 3 と反応 2 が起こり、反応器出口で平衡が達成されるとして出口の各成分の分率を求めよ。ただし、反応器条件は出口で温度 250°C、7.5 MPa とする。

Before starting to write the program to find the molar fractions of the components, it is necessary to first determine the limits or boundaries of the flow of number of moles that changes for reaction three (n3) and for reaction two (n2).

(reaction 3)	CO	+	2 H_2	→	CH_3OH
equilibrium state	F0CO-n3-n2		F0H_2-2*n3+n2		F0CH_3OH+n3

(reaction 2)	CO	+	H_2O	→	CO_2	+	H_2
equilibrium state	F0CO-n3-n2		F0H_2O -n2		F0CO_2 +n2		F0H_2-2*n3+n2

F0X is used to symbolize the flow of component X (= CO, H_2, CH_3OH, H_2O, CO_2) for an initial state.

Reactions three and two share the variables n3 and n2 within the equilibrium equations, therefore we must propose a system of inequalities, in which we will find a feasible region for n3 and n2.

$0 \leq F0CO-n3-n2 \rightarrow n3 \leq F0CO-n2$

$0 \leq F0H_2-2*n3+n2 \rightarrow n3 \leq (F0H_2+n2)/2$

$0 \leq F0CH_3OH+n3 \rightarrow n3 \geq -F0CH_3OH$

$0 \leq F0H_2O -n2 \rightarrow n2 \leq F0H_2O$

$0 \leq F0CO_2 +n2 \rightarrow n2 \geq -F0CO_2$

but according to the problem the initial conditions of the mol flows are the following:

$F0CH_4$=109.6 kmol/h
$F0H_2O$=13.4 kmol/h
F0CO=753.4 kmol/h
$F0H_2$=2808.3 kmol/h
$F0CO_2$=137.0 kmol/h
$F0CH_3OH$=0 kmol/h

F0X is used to symbolize the flow of component X (= CO, H_2, CH_3OH, H_2O, CO_2) for an initial state.
Under these conditions and inequalities I plot the feasible limits of initial n2 and n3 as shown in Figure 1 (for this the following code was used). y symbolizes n3 and x symbolizes n2.

CODE

```
import numpy as np
import matplotlib.pyplot as plt
import math

F0CH4=109.6
F0H2O=13.4
F0CO=753.4
F0H2=2808.3
F0CO2=137.0
F0CH3OH=0

plt.figure(figsize=(8,8))
plt.grid()
# plot the feasible region
d = np.linspace(-150,900,300)
x,y = np.meshgrid(d,d)
plt.imshow( ((y<=F0CO-x)&(y<=(F0H2+x)/2) & (y>=0) & (x<=F0H2O) & (x>=-
F0CO2)).astype(int) ,
            extent=(x.min(),x.max(),y.min(),y.max()),origin="lower", cmap="Greys", alpha = 0.3);

# plot the lines defining the constraints
x = np.linspace(-2850, 800, 2000)

y1=F0CO-x
plt.plot(x, y1, label=r'$y\leq F0CO-x$')

y2=(F0H2+x)/2
plt.plot(x, y2, label=r'$y\leq (F0H2+x)/2$')

y3=0*x
plt.plot(x, y3,color='k', label=r'$y\geq 0$')

plt.axvline(x=F0H2O,colorThird, to express the molar flow of each component for each stage of the process, we will use the notation FYX, where Y (= 1, 2, 3, 4, 5, 6, 7, 8) shows the stage where the components are found. X (= CO, H2, CH3OH, H2O, CO2) shows a specific component portion.='r',label=r'$x\leq F0H2O$')

plt.axvline(x=-F0CO2,color='g',label=r'$x\geq -F0CO2$')

# Make plot
plt.title('Boundaries of n02 and n03 ', fontsize=25)
plt.xlim(-500,800)
plt.ylim(-50,1300)
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.,fontsize=15)
plt.xlabel('n02 [kmol/h]= x axis',fontsize=20)
```

plt.ylabel(' n03 [kmol/h] =y axis',fontsize=20)

plt.show()

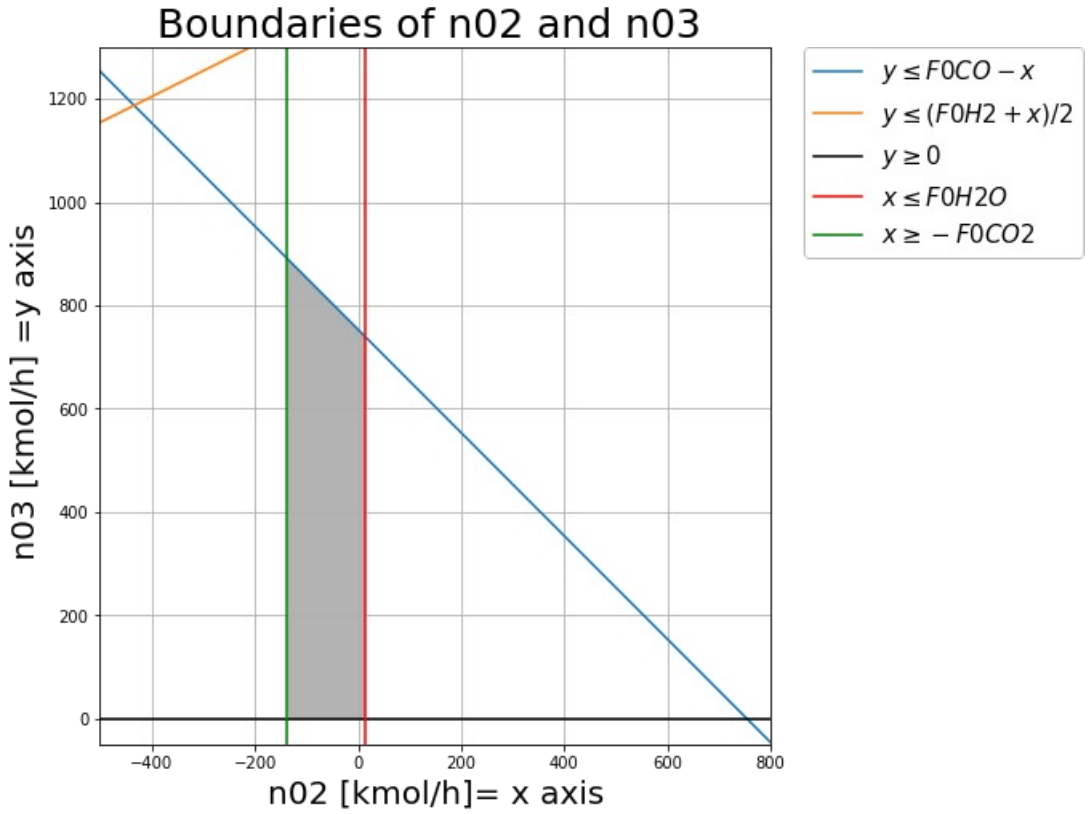

Figure 1: Boundaries of n02 and n03

As seen in Figure 1, the limits of n2 are determined by the by the green and red lines. But for the case of n3, its lower limit is greater than or equal to zero, but its upper limit is determined by the blue line of equation y≤F0CO-x. Therefore the upper limit of n3 is dependent on n2.
Then Boundaries of n02 and n03:

-F0CO2≤n2≤F0H20
0≤n3≤F0CO-n2

Knowing these important limits, I can create functions to find the mole fraction fluxes of each component using 二分法 to compare the values of the real equilibrium constant Kp2, Kp3 versus the calculated equilibrium constants Kp3Calc and Kp2Calc.

First I am going to create a function 'FunctionEquilibrium (T, P)' which depends on the reaction temperature T [℃] and reaction pressure P [Pa]. Inside of the function Te symbolizes reaction temperature in Kelvin grades and Pe symbolizes reaction pressure in atm. Then Te was used to find Kp3 and Kp2 values according to the following equation:

$Kp3=\exp(21.225+9143.6/Te - 7.492\ln Te + 4.076\times 10^{-3} Te - 7.161\times 10^{-8} Te^2)$

$Kp2 = \exp(13.148 - 5639.5/Te - 1.077\ln Te - 5.44\times 10^{-4} Te + 1.125\times 10^{-7} Te^2 + 49170/Te^2)$

Second, Using the initial conditions of molar flow F0 I express the limits of n2.

```
n2min=-F0CO2
n2max=F0H2O
```

Third, I will use two while loops, the first to approximate the value of n2 as (n2max + n2min) / 2. This first loop will be repeated as many times as necessary under the n3max-n3min>0 condition. The second loop is inside the n2 loop, it starts when the value of n2 is set, and its maximum and minimum limits are taken into account as n3min=0, n3max=F0CO-n2. In the same way as n2, a first n3 = (n3max + n3min) / 2 is established, which will be used to calculate the equilibrium equations of Kp2Calc and Kp3Calc. Their equations are expressed as:

```
FCO=F0CO-n3-n2
FH2O=F0H2O-n2
FCO2=F0CO2+n2
FH2=F0H2-2*n3+n2
FCH3OH=F0CH3OH+n3
FCH4=F0CH4
Ftotal=FCH4+FH2O+FCO+FH2+FCO2+FCH3OH

yCH4=FCH4/Ftotal
yH2O=FH2O/Ftotal
yCO=FCO/Ftotal
yH2=FH2/Ftotal
yCO2=FCO2/Ftotal
yCH3OH=FCH3OH/Ftotal

Kϕ2=1.0
Kp2Calc=yCO2*yH2/(yCO*yH2O)*Kϕ2

Kϕ3=1.0
Kp3Calc=yCH3OH/(yCO*yH2**2)/Pe**2*Kϕ3
```

Here FX is used to symbolize the flow of component X (= CO, H_2, CH_3OH, H_2O, CO_2) for equilibrium state. Also yX is used to symbolize the molar fractions of the components X (= CO, H_2, CH_3OH, H_2O, CO_2) for equilibrium state. Ftotal shows the total flow of the whole components. Pe symbolizes reaction pressure in atm. Kϕ2 and Kϕ3 are Activity coefficients.

Then these values of equilibrium constants Kp2Calc and Kp3Calc will be compared with the real values of Kp2 and Kp3 by means of if and else logical functions. These two loops will end when the values of n2max-n2min and n3max-n3min are less than 10^{-6}.

Finally all these calculated valued are printed according to:

```
print("Kp3=",Kp3)
print("Kp3Calc=",Kp3Calc)
print("Kp2=",Kp2)
print("Kp2Calc=",Kp2Calc)
print("n3=",n3,"mol/h")
print("n2=",n2,"mol/h")
print("yCH4=",yCH4)
print("yH2O=",yH2O)
print("yCO=",yCO)
print("yH2=",yH2)
print("yCO2=",yCO2)
```

```
    print("yCH3OH=",yCH3OH)
```

CODE

```
import numpy as np
import matplotlib.pyplot as plt
import math

def FunctionEquilibrium(T,P):

    Te=T+273.15
    Pe=P/101325

    Kp3=math.exp(21.225+9143.6/Te-7.492*math.log(Te)+4.076*10**-3*Te-7.161*10**-8*Te**2)
    Kp2=math.exp(13.148-5639.5/Te-1.077*math.log(Te)-5.44*10**-4*Te+1.125*10**-
7*Te**2+49170/Te**2)

    F0CH4=109.6*10**3
    F0H2O=13.4*10**3
    F0CO=753.4*10**3
    F0H2=2808.3*10**3
    F0CO2=137.0*10**3
    F0CH3OH=0
    F0TOTAL=F0CH4+F0H2O+F0CO+F0H2+F0CO2+F0CH3OH

    n2min=-F0CO2
    n2max=F0H2O

    while n2max-n2min>10**-6:
        n2=(n2min+n2max)/2.0

        n3min=0
        n3max=F0CO-n2

        while n3max-n3min>10**(-6):
            n3=(n3max+n3min)/2

            FCO=F0CO-n3-n2
            FH2O=F0H2O-n2
            FCO2=F0CO2+n2
            FH2=F0H2-2*n3+n2
            FCH3OH=F0CH3OH+n3
            FCH4=F0CH4
            Ftotal=FCH4+FH2O+FCO+FH2+FCO2+FCH3OH

            yCH4=FCH4/Ftotal
            yH2O=FH2O/Ftotal
            yCO=FCO/Ftotal
            yH2=FH2/Ftotal
            yCO2=FCO2/Ftotal
            yCH3OH=FCH3OH/Ftotal
```

```
            Kϕ2=1.0
            Kp2Calc=yCO2*yH2/(yCO*yH2O)*Kϕ2
            Kϕ3=1.0
            Kp3Calc=yCH3OH/(yCO*yH2**2)/Pe**2*Kϕ3

            if Kp3Calc>Kp3:
               n3max=n3
            else:
               n3min=n3

         if Kp2Calc>Kp2:
            n2max=n2
         else:
            n2min=n2

      print("Kp3=",Kp3)
      print("Kp3Calc=",Kp3Calc)
      print("Kp2=",Kp2)
      print("Kp2Calc=",Kp2Calc)
      print("n3=",n3,"mol/h")
      print("n2=",n2,"mol/h")
      print("yCH4=",yCH4)
      print("yH2O=",yH2O)
      print("yCO=",yCO)
      print("yH2=",yH2)
      print("yCO2=",yCO2)
      print("yCH3OH=",yCH3OH)

FunctionEquilibrium(250,7.5*10**6)
```

Results

Kp3= 0.0022808615438347207
Kp3Calc= 0.0022808615438381676
Kp2= 0.011702947213532435
Kp2Calc= 0.011702947219362441
n3= 691772.1094997011 mol/h
n2= -136729.42694177618 mol/h
yCH4= 0.04495200874942499
yH2O= 0.06157499373570149
yCO= 0.08135547325883788
yH2= 0.5282789410323186
yCO2= 0.00011097447518828214
yCH3OH= 0.2837276087485287

How do I know if these mole fraction values are correct? I can know this thanks to the values of Kp2Calc and Kp3Calc, which are very close to the real values of Kp2 and Kp3. In conclusion I was able to successfully find the values of the mole fractions for each component in the equilibrium state

2-2) r をリサイクルフローの比率(リサイクルされるガス流量の原料ガス流量に対する比率)、p をパージの比率(パージされるガス流量のリサイクルされるガス流量に対する比率)とする。

2-1)と同様に合成ガスを供給した場合の、r の変化に対する p、合成されるメタノールの流量、の変化を図示せよ。ただし、フラッシュ分離では液化した CH 3 OH、H 2 O に他の成分は溶解しないと仮定する。

実際のプラントでは r は 7.8 程度に設定さている。その理由について考察せよ.

I first find the vapor pressures of methanol PCH3OHvap [Pa] and water PH2Ovap [Pa]. For this we use Antoine's equation with their respective numerical constants A, B and C; for a Flash separation temperature Ts = 35 + 273.15 [° C]. Also the total pressure of the system for this operation is Ptotal = 7.5 * 10^6 [Pa]

PCH3OHvap=exp(ACH3OH-BCH3OH/(CCH3OH+Ts))
PH2Ovap=exp(AH2O-BH2O/(CH2O+Ts))

Second, I create the “FunctionEquilibrium (T, P, p)” function, like the previous problem, this depends on the reaction temperature T [° C] and reaction pressure P [Pa]. Inside of the function Te symbolizes reaction temperature in Kelvin grades and Pe symbolizes reaction pressure in atm; thought these values I calculated Kp3 and Kp2. In addition a new variable "p" is added which shows Purge ratio.

$Kp3=\exp(21.225+9143.6/Te-7.492\ln Te+4.076\times 10^{-3} Te-7.161\times 10^{-8} Te^2)$

$Kp2=\exp(13.148-5639.5/Te-1.077\ln Te-5.44\times 10^{-4} Te+1.125\times 10^{-7} Te^2+49170/Te^2)$

Third, to express the molar flow of each component for each stage of the process, we will use the notation FYX, where X (= CO, H2, CH3OH, H2O, CO2) shows a specific component portion. Y (= 1, 2, 3, 4, 5, 6, 7, 8) shows the stage where the components are found. Y is declared as written in Figure 2.

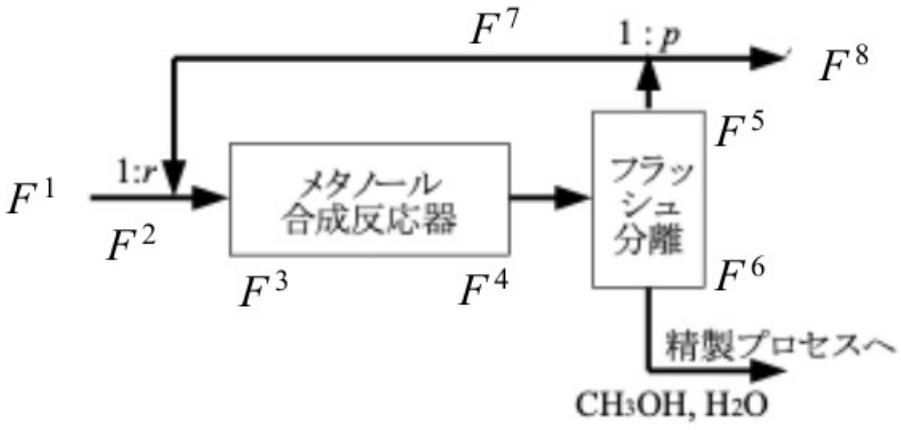

Figure 2: Position of molar flows for different stages

I am not going to use F1 at all, it is assumed that this is the same as F2. F2 molar flows have the same initial composition as in the previous problem. In addition Fi symbolizes the iterative molar flows for the different components. By condition of the problem this is equal to 0 for all Fi.

F2CH4=109.6*10^3 mol/h
F2H2O=13.4*10^3 mol/h
F2CO=753.4*10^3mol/h
F2H2=2808.3*10^3 mol/h
F2CO2=137.0*10^3 mol/h
F2CH3OH=0 mol/h

Fi7CH4=0 mol/h
Fi7H2O=0 mol/h
Fi7CO=0 mol/h
Fi7H2=0 mol/h
Fi7CO2=0 mol/h
Fi7CH3OH=0 mol/h

Like the previous problem, I will use loops with the name "while". The first "while True" will serve to converge the value of Fi7total. Also, as shown in Figure 2, F3 is the sum of the molar flows F2 and F7, but I will use Fi7 instead of F7 since this will help me find the value of the true F7.

Fi7total=Fi7CH4+Fi7CO+Fi7CO2+Fi7H2+Fi7H2O+Fi7CH3OH
bascis
F3CH4=Fi7CH4+F2CH4
F3H2O=Fi7H2O+F2H2O
F3CO=Fi7CO+F2CO
F3H2=Fi7H2+F2H2
F3CO2=Fi7CO2+F2CO2
F3CH3OH=Fi7CH3OH+F2CH3OH

What follows is exactly the same as the previous problem with the two "while" loops. The only difference is that F and F0 from the previous problem become F4 and F3 molar fluxes respectively. In addition, "y" from the above problem becomes "y4" mole fraction. Despite the redundancy, it is necessary to use well the limits of molar flows n2 and n3. Since n3 depends on n2. (look problem number 2-1) for more detailed explanation about this last two while loops).

n2min=-F3CO2
n2max=F3H2O

n3min=0
n3max=F3CO-n2

We are still inside the "while True" loop. After the F4 is obtained, it goes through an フラッシュ分離 process, obtaining F5 for the volatile part and F6 for the heavy part which only contains methanol and water. To obtain equations of F5H2O and F5CH3OH I used the following relation:

PH2Ovap : PCH3OHvap： P total =
F5H2O : F5CH3OH : (F5H2O+F5CH3OH+F5CH4+F5CO+F5CO2+F5H2)

F5CH4=F4CH4
F5CO=F4CO
F5CO2=F4CO2
F5H2=F4H2
F5H2O=(F5CH4+F5CO+F5CO2+F5H2)*PH2Ovap/(Ptotal-PH2Ovap-PCH3OHvap)

F5CH3OH=(F5CH4+F5CO+F5CO2+F5H2)*PCH3OHvap/(Ptotal-PH2Ovap-PCH3OHvap)

F6CH4=0
F6CO=0
F6CO2=0
F6H2=0
F6H2O=F4H2O-F5H2O
F6CH3OH=F4CH3OH-F5CH3OH

As seen in Figure 2, the flow F5 is equal to the sum of F7 and F8. Furthermore there is a relation of F7: F8 = 1: p, where p is the purge rate.

F7CH4=F5CH4/(1+p)
F7CO=F5CO/(1+p)
F7CO2=F5CO2/(1+p)
F7H2=F5H2/(1+p)
F7H2O=F5H2O/(1+p)
F7CH3OH=F5CH3OH/(1+p)
F7total=F7CH4+F7CO+F7CO2+F7H2+F7H2O+F7CH3OH

F8CH4=F5CH4*p/(1+p)
F8CO=F5CO*p/(1+p)
F8CO2=F5CO2*p/(1+p)
F8H2=F5H2*p/(1+p)
F8H2O=F5H2O*p/(1+p)
F8CH3OH=F5CH3OH*p/(1+p)

From this total F7, I compare it with the first total Fi7. For the system to be in a steady state. F7 must be the same as Fi7. For that I use the "if" condition

if abs (F7total-Fi7total)> 10^{-6}

If this condition is fulfilled then F7 is updated or converted to Fi7 as shown in the following equations:

Fi7CH4=F7CH4
Fi7H2O=F7H2O
Fi7CO=F7CO
Fi7H2=F7H2
Fi7CO2=F7CO2
Fi7CH3OH=F7CH3OH

Then the "while True" loop will start over with a new Fi7. This loop will end when the "else" case occurs. I mean when
abs (F7total-Fi7total)<10^{-6}

Then I can calculate r, the reclying flow rate as

r=(F7CH4+F7CO+F7CO2+F7H2+F7H2O+F7CH3OH)/(F2CH4+F2CO+F2CO2+F2H2+F2H2O+F2CH3OH)

Finall, I print The following results FunctionEquilibrium(250,7.5*10**6,0.01), p=0.01

```
        print("Kp3=",Kp3)
        print("Kp3Calc=",Kp3Calc)
        print("Kp2=",Kp2)
        print("Kp2Calc=",Kp2Calc)
        print("n3=",n3,"mol/h")
        print("n2=",n2,"mol/h")
        print("p=",p)
        print("F6CH3OH=",F6CH3OH)
        print("r=",r)

        return p,F6CH3OH,r

        break
```

FunctionEquilibrium(250,7.5*10**6,0.01)

COMPLETE CODE

```
import numpy as np
import matplotlib.pyplot as plt
import math

Ts=35+273.15
ACH3OH = 23.48027002
BCH3OH = 3626.55
CCH3OH = -34.29
PCH3OHvap=math.exp(ACH3OH-BCH3OH/(CCH3OH+Ts))
AH2O = 23.19637
BH2O = 3816.44
CH2O = -46.13
PH2Ovap=math.exp(AH2O-BH2O/(CH2O+Ts))
Ptotal=7.5*10**6
print("PCH3OHvap=",PCH3OHvap,"Pa")
print("PH2Ovap=",PH2Ovap,"Pa")

def FunctionEquilibrium(T,P,p):

    Te=T+273.15
    Pe=P/101325

    Kp3=math.exp(21.225+9143.6/Te-7.492*math.log(Te)+4.076*10**-3*Te-7.161*10**-8*Te**2)
    Kp2=math.exp(13.148-5639.5/Te-1.077*math.log(Te)-5.44*10**-4*Te+1.125*10**-
7*Te**2+49170/Te**2)

    F2CH4=109.6*10**3
    F2H2O=13.4*10**3
    F2CO=753.4*10**3
    F2H2=2808.3*10**3
    F2CO2=137.0*10**3
    F2CH3OH=0
```

```
Fi7CH4=0
Fi7H2O=0
Fi7CO=0
Fi7H2=0
Fi7CO2=0
Fi7CH3OH=0

while True:
    Fi7total=Fi7CH4+Fi7CO+Fi7CO2+Fi7H2+Fi7H2O+Fi7CH3OH

    F3CH4=Fi7CH4+F2CH4
    F3H2O=Fi7H2O+F2H2O
    F3CO=Fi7CO+F2CO
    F3H2=Fi7H2+F2H2
    F3CO2=Fi7CO2+F2CO2
    F3CH3OH=Fi7CH3OH+F2CH3OH

    n2min=-F3CO2
    n2max=F3H2O

    while n2max-n2min>10**(-6):
        n2=(n2min+n2max)/2

        n3min=0
        n3max=F3CO-n2

        while n3max-n3min>10**(-6):
            n3=(n3max+n3min)/2

            F4CO=F3CO-n3-n2
            F4H2O=F3H2O-n2
            F4H2=F3H2-2*n3+n2
            F4CH3OH=F3CH3OH+n3
            F4CO2=F3CO2+n2
            F4CH4=F3CH4
            F4total=F4CH4+F4H2O+F4CO+F4H2+F4CO2+F4CH3OH

            y4CH4=F4CH4/F4total
            y4H2O=F4H2O/F4total
            y4CO=F4CO/F4total
            y4H2=F4H2/F4total
            y4CO2=F4CO2/F4total
            y4CH3OH=F4CH3OH/F4total

            Kϕ3=1.0
            Kp3Calc=y4CH3OH/(y4CO*y4H2**2)/Pe**2*Kϕ3

            Kϕ2=1.0
            Kp2Calc=y4CO2*y4H2/(y4CO*y4H2O)*Kϕ2

            if Kp3Calc>Kp3:
```

```
            n3max=n3
        else:
            n3min=n3

    if Kp2Calc>Kp2:
        n2max=n2
    else:
        n2min=n2

F5CH4=F4CH4
F5CO=F4CO
F5CO2=F4CO2
F5H2=F4H2
F5H2O=(F5CH4+F5CO+F5CO2+F5H2)*PH2Ovap/(Ptotal-PH2Ovap-PCH3OHvap)
F5CH3OH=(F5CH4+F5CO+F5CO2+F5H2)*PCH3OHvap/(Ptotal-PH2Ovap-PCH3OHvap)

F6CH4=0
F6CO=0
F6CO2=0
F6H2=0
F6H2O=F4H2O-F5H2O
F6CH3OH=F4CH3OH-F5CH3OH

F7CH4=F5CH4/(1+p)
F7CO=F5CO/(1+p)
F7CO2=F5CO2/(1+p)
F7H2=F5H2/(1+p)
F7H2O=F5H2O/(1+p)
F7CH3OH=F5CH3OH/(1+p)
F7total=F7CH4+F7CO+F7CO2+F7H2+F7H2O+F7CH3OH

F8CH4=F5CH4*p/(1+p)
F8CO=F5CO*p/(1+p)
F8CO2=F5CO2*p/(1+p)
F8H2=F5H2*p/(1+p)
F8H2O=F5H2O*p/(1+p)
F8CH3OH=F5CH3OH*p/(1+p)

if abs(F7total-Fi7total)>10**-6:
    Fi7CH4=F7CH4
    Fi7H2O=F7H2O
    Fi7CO=F7CO
    Fi7H2=F7H2
    Fi7CO2=F7CO2
    Fi7CH3OH=F7CH3OH

else:

r=(F7CH4+F7CO+F7CO2+F7H2+F7H2O+F7CH3OH)/(F2CH4+F2CO+F2CO2+F2H2+F2H2O+F
2CH3OH)

    print("Kp3=",Kp3)
```

```
        print("Kp3Calc=",Kp3Calc)
        print("Kp2=",Kp2)
        print("Kp2Calc=",Kp2Calc)
        print("n3=",n3,"mol/h")
        print("n2=",n2,"mol/h")
        print("p=",p)
        print("F6CH3OH=",F6CH3OH,"mol/h")
        print("r=",r)
        print("F7total=",F7total,"mol/h")
        print("Fi7total=",Fi7total,"mol/h")
        return p,F6CH3OH,r

        break

FunctionEquilibrium(250,7.5*10**6,0.01)
```

RESULTS

PCH3OHvap= 27942.941714247885 Pa
PH2Ovap= 5602.213021928711 Pa
Kp3= 0.0022808615438347207
Kp3Calc= 0.002280861543852358
Kp2= 0.011702947213532435
Kp2Calc= 0.011702947563299782
n3= 889095.9790590436 mol/h
n2= -136999.9618111735 mol/h
p= 0.01
F6CH3OH= 885338.4994786554 mol/h
r= 26.389379297544643
F7total= 100852290.86142637 mol/h
Fi7total= 100852290.86142565 mol/h
(0.01, 885338.4994786554, 26.389379297544643)

So, from this result, I can verify that the code worked well, since the Kp3Calc, Kp2Calc and F7total values converge to the Kp3, Kp2 and Fi7total values.

Next, I am going to use the recently created function FunctionEquilibrium (250,7.5 * 10 ** 6, p), for "p" that varies between (0.01, 0.02, ... 0.10) to find the values of "F6CH3OH" and "r" respective.

Purge value is equal to FunctionEquilibrium(250,7.5*10**6,p)[0]
F6CH3OH value is equal to FunctionEquilibrium(250,7.5*10**6,p)[1]
Recycle flow ratio value is equal to FunctionEquilibrium(250,7.5*10**6,p)[2]

All this I do as shown in the following plotting codes:

CODE for F6CH3OH vs Recycle flow ratio figure 3

```
import numpy as np
import matplotlib.pyplot as plt
import math
```

```
i=0
r=[]
F6CH3OH=[]
while i<10:
    i=i+1
    p=i/100

    r.append(FunctionEquilibrium(250,7.5*10**6,p)[2])
    F6CH3OH.append(FunctionEquilibrium(250,7.5*10**6,p)[1]/1000)

plt.figure(figsize=(8,8))
plt.plot(r,F6CH3OH,'bo')

plt.grid()
plt.title('F6CH3OH vs Recycle flow ratio', fontsize=25)
plt.xlabel('Recycle flow ratio r [-]', fontsize=20)
plt.ylabel('F6CH3OH [kmol/h]', fontsize=20)
plt.show()
```

CODE for 'Purge ratio vs Recycle flow ratio' figure 4

```
import numpy as np
import matplotlib.pyplot as plt
import math

i=0
r=[]
purge=[]
while i<10:
    i=i+1
    p=i/100

    r.append(FunctionEquilibrium(250,7.5*10**6,p)[2])
    purge.append(FunctionEquilibrium(250,7.5*10**6,p)[0])

plt.figure(figsize=(8,8))
plt.plot(r,purge,'bo')

plt.grid()
plt.title('Purge ratio vs Recycle flow ratio', fontsize=25)
plt.xlabel('Recycle flow ratio r [-]', fontsize=20)
plt.ylabel('Purge ratio [-]', fontsize=20)
plt.show()
```

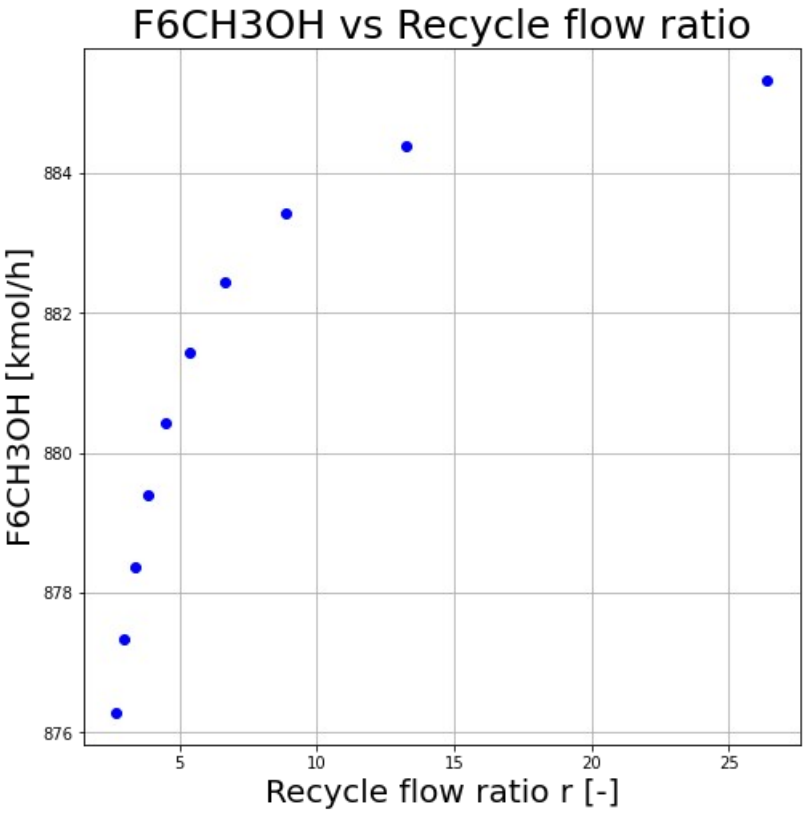

Figure 3: F6CH3OH vs Recycle flow ratio

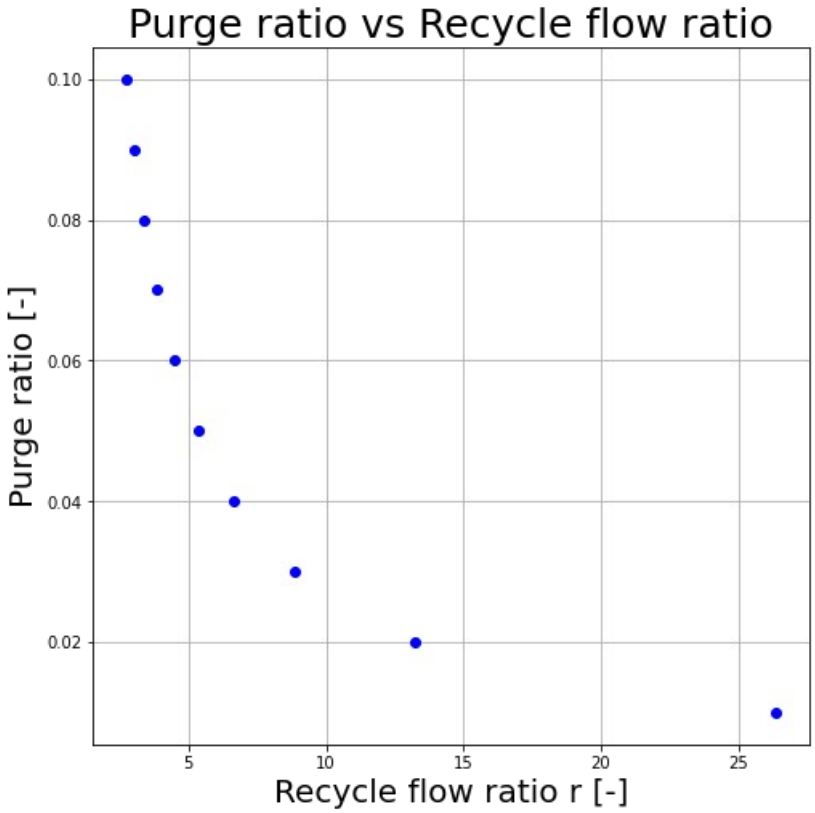

Figure 4: Purge ratio vs Recycle flow ratio

The data in Figures 3 and 4 can be grouped in Table 1.

Table 1: p vs r vs F6CH3OH

p [-]	r [-]	F6CH3OH [kmol/h]
0.01	26.3894	885.3385
0.02	13.2318	884.3875
0.03	8.8463	883.4195
0.04	6.6539	882.4359
0.05	5.3387	881.4380
0.06	4.4621	880.4272
0.07	3.8360	879.4048
0.08	3.3666	878.3720
0.09	3.0015	877.3299
0.10	2.7096	876.2797

As can be seen in figure 3, the methanol F6CH3OH obtained grows when r increases. However, there is not much methanol growth for an r greater than 10. So that so much energy is not spent in the process, it would be advisable and efficient for r to be between 5 and 10. From figure 4 we observe that there are "purge "Very large for r less than 7. Then it can be concluded that a good r for this system would be one that is between the values of 7 and 10, to be more precise a value of r = 7.8 would give us a very efficient process for a low purge and a good yield of methanol F6CH3OH.

Here I share my code on the github platform.

https://github.com/marcelo1356/-1-3/blob/main/PROCESS%20HOMEWOK%202.ipynb